CLAIMS

1. Code generator with a plurality of storage elements ($FF_{1,2,...n}$) connected in a code-producing series (R), e.g., flip-flops, wherein the output of the last storage element (FF_5) in the series (R) is linked with the input of the first storage element (FF_1) in the series (R) to form a circuit, and outputs and inputs of the storage elements are recursively interconnected with EXOR gates inserted, characterized in that at least one EXOR gate ($EXOR_{p1}$) is provided, whose first input (1) is connected with the output of a storage element (FF_1) located in the code-producing series (R), whose second input (2) is connected with the output of another storage element (FF_3) located in the code-producing series(R), and whose output (3) is connected with the input of the storage element (FF_2) following the storage element (FF_1) connected with the first input (1) of the EXOR gate ($EXOR_{p1}$) in the code-producing series (R), and that the output of a storage element (FF_5) located in the code-producing series (R) is connected with the input of an inverter (INV), and the output of the inverter (INV) is connected with the input of another storage element (FF_1) arranged in the code-producing series (R).

2. Code generator according to claim 1, characterized in that an AND gate (AND_{p1}) is connected in the line connecting the second input (2) of the at least one EXOR gate ($EXOR_{p1}$) and the output of the other storage element (FF_3) located in the code-reproducing series (R), so that the output (4) of the AND gate (AND_{p1}) is connected with the second input (2) of the EXOR gate ($EXOR_{p1}$), the first input (6) of the AND gate (AND_{p1}) is connected with the output of the other storage element (FF_3) located in the code-producing series (R), and the second input (5) of the AND gate (AND_{p1}) is connected with the output of a code-programming storage element (FF_{p1}).

3. Code generator according to claim 1 or 2, characterized in that a plurality of EXOR gates ($EXOR_{p1,p2,p3,p4}$) is provided, whose first input is supplied by a respective output of one of the storage elements ($FF_{1,2,3,4}$) located in the code-producing series (R), and whose second input is supplied by the respective output of another storage element ($FF_{8,15,20,23}$) located in the code-producing series (R), which is spaced a number of storage elements in the conducting direction of the series (R) away from the storage element ($FF_{1,2,3,4}$) respectively connected with the first input, which respectively corresponds to

a different prime number that is greater than 1 and does not constitute a partial amount of the overall number of storage elements ($FF_{1,2,...n}$) connected in series (R).

4. Code generator according to claim 1, 2 or 3, characterized in that a plurality of code-programming storage elements ($FF_{p1,p2,p3,p4,...pn}$) that are respectively assigned to an AND gate ($AND_{p1,p2,p3,p4}$) and an EXOR gate ($EXOR_{p1,p2,p3,p4}$) is provided and connected in a series (RR) comprising a closed circuit, and at least one EXOR gate ($EXOR_{pp1}$) is provided whose first input is connected with the output of a storage element (FF_{p6}) located in the code-programming series (RR), whose second input is connected with the output of another storage element (FF_{p5}) located in the code-programming series (RR), and whose output is connected with the input of the storage element (FF_{p1}) following the storage element (FF_{p5}) connected with the first input of the EXOR gate ($EXOR_{pp1}$) in the code-programming series (RR).

5. Code generator according to claim 4, characterized in that an AND gate (AND_{pp1}) is connected in the line connecting the second input of the at least one EXOR gate ($EXOR_{pp1}$) and the output of the other storage element (FF_{p3}) located in the code-reproducing series (RR), so that the output of the AND gate (AND_{pp1}) is connected with the second input of the EXOR gate ($EXOR_{pp1}$), the first input of the AND gate (AND_{pp1}) is connected with the output of the other storage element (FF_{p3}) located in the code-producing series (RR), and the second input of the AND gate (AND_{pp1}) is connected with the output of a storage element (FF_{pp5}) used for programming the code-programming series (RR).

6. Code generator according to claim 5, characterized in that a plurality of storage elements ($FF_{pp1,pp2,pp3,pp4,...ppn}$) used to program the code-programming series (RR) that are respectively assigned to an AND gate (AND_{pp1}) and an EXOR gate ($EXOR_{pp1}$) is provided and connected in a series (RRR) comprising a closed circuit, and at least one EXOR gate ($EXOR_{ppp1}$) is provided whose first input is connected with the output of a storage element (FF_{pp1}) located in the series (RRR), whose second input is connected with the output of another storage element (FF_{pp3}) located in the series (RRR), and whose output is connected with the input of the storage element (FF_{pp2})

following the storage element (FF_{pp1}) connected with the first input of the EXOR gate ($EXOR_{ppp1}$) in the series (RRR).

7. Code generator according to one of claims 1 to 6, characterized in that it has at least one connection for at least a second, identically structured code generator, so that both code generators can be supplied with the same program clock at the same time.

8. Device for sending and receiving encrypted information with at least two code generators according to one of claims 1 to 7, characterized in that the code generators each have at least one connection for simultaneously supplying the code-programming storage elements ($FF_{p1,p2,p3,p4}$) of all interconnected code generators with the same program clock, so that the code-programming storage elements ($FF_{p1,p2,p3,...pn}$) of all interconnected code generators simultaneously run through all possible state combinations, and are provided with the same programming when the code generators are simultaneously separated from the program clock.

9. Device according to claim 8, characterized in that the code generators each have two connections for simultaneously supplying the code-programming storage elements ($FF_{p1,p2,p3,...pn}$) and the storage elements ($FF_{pp1,pp2,pp3,...ppn}$) used to program the code-programming storage elements ($FF_{p1,p2,p3,...pn}$) of all interconnected code generators have two independently running program clocks, wherein the storage elements ($FF_{pp1,pp2,pp3,...ppn}$) used to program the code-programming storage elements ($FF_{p1,p2,p3,...pn}$) run through all possible state combinations at least once, and the code-programming storage elements ($FF_{p1,p2,p3,...pn}$) of all interconnected code generators simultaneously run through a specific number of all possible state combinations, and that all interconnected code generators are provided with the same programming after the simultaneous separation of code generators from the program clocks.